Online Shopping Platform

SQL ASSIGNMENT

-By Akshay Gujare
Mob no: 7741841945
77418analyst@gmail.com

ONLINE SHOPPING PLATFORM

SQL Assignment

Now, to get a better understanding we will be extending the RetailDB. So, here we'll work with **5 tables**:

1. **Customers** – customer details
   (customer_id, name, email, city, signup_date**)**

2. **Orders** – order details
   (order_id, customer_id, order_date, total_amount, payment_mode**)**

3. **Products** – product details
   (product_id, product_name, category, price, stock_qty**)**

4. **Order_Items** – links orders with products
   (order_item_id, order_id, product_id, quantity, price_each**)**

5. **Suppliers** – product suppliers
   (supplier_id, supplier_name, contact_email, city**)**

Customers

customer_id	name	email	city	signup_date
1	Rohit Kumar	rohit.kumar@gmail.com	Delhi	2024-01-15
2	Sneha Sharma	sneha.sharma@yahoo.com	Mumbai	2024-02-20
3	Amit Patel	amit.patel@gmail.com	Ahmedabad	2024-03-05
4	Priya Reddy	priya.reddy@gmail.com	Hyderabad	2024-03-22
5	Karan Singh	karan.singh@outlook.com	Chennai	2024-04-10
6	Neha Verma	neha.verma@gmail.com	Pune	2024-05-08

Orders

order_id	customer_id	order_date	total_amount	payment_mode
1	1	2024-07-10	82498.00	UPI
2	2	2024-07-15	74999.00	Credit Card
3	3	2024-07-20	5498.00	Net Banking
4	4	2024-07-22	5998.00	Debit Card
5	5	2024-08-01	1999.00	UPI
6	6	2024-08-03	6499.00	Credit Card
7	1	2024-08-05	1499.00	UPI
8	2	2024-08-07	59999.00	Debit Card
9	3	2024-08-10	1497.00	UPI
10	4	2024-08-12	55999.00	Net Banking
11	5	2024-08-15	2999.00	Credit Card
12	6	2024-08-18	285.00	UPI
13	1	2024-08-20	499.00	Cash
14	2	2024-08-22	1999.00	UPI
15	3	2024-08-25	55999.00	Net Banking

Products

product_id	product_name	category	price	stock_qty	supplier_id
1	iPhone 15	Electronics	79999.00	25	1
2	Samsung Galaxy S24	Electronics	74999.00	30	2
3	Noise Smartwatch	Wearables	4999.00	100	2
4	Boat Earbuds	Wearables	2499.00	150	3
5	Kurta Set	Fashion	1999.00	200	3
6	Running Shoes	Fashion	2999.00	180	3
7	Prestige Mixer Grinder	Home Appliances	6499.00	75	4
8	Tata Tea 1kg Pack	Groceries	499.00	300	4
9	Amul Butter 500g	Groceries	285.00	400	4
10	Sony Bravia 55" TV	Electronics	59999.00	20	1
11	Lenovo Laptop	Electronics	55999.00	40	1
12	Philips Trimmer	Personal Care	1499.00	120	2

Order_Items

order_item_id	order_id	product_id	quantity	price_each
1	1	1	1	79999.00
2	1	4	1	2499.00
3	2	2	1	74999.00
4	3	3	1	4999.00
5	3	6	1	2999.00
6	4	5	3	1999.00
7	5	7	1	6499.00
8	6	12	1	1499.00
9	7	9	2	285.00
10	8	10	1	59999.00
11	9	8	3	499.00
12	10	11	1	55999.00
13	11	6	1	2999.00
14	12	9	1	285.00
15	13	8	1	499.00
16	14	5	1	1999.00
17	15	11	1	55999.00

Suppliers

supplier_id	supplier_name	contact_email	city
1	Reliance Digital	contact@reliancedigital.in	Mumbai
2	Croma Electronics	support@croma.com	Bengaluru
3	Big Bazaar Online	sales@bigbazaar.com	Delhi
4	DMart Suppliers	dmart@dmart.in	Pune

Instructions to Load the tables on MySQL Workbench:

1. *Download the files from folder link:*

https://drive.google.com/drive/folders/1fHOzGXkzKlOwmnLsiecWha4iQYlxf1qE?usp=drive_link

2. Create the database with name as - ***RetailDB_2***

3. *Load the files on the created tables in MySQL Workbench using fetch csv data.*

4. *Can view with select command.*

Task: Solve all the below mentioned problems by writing the SQL queries.

a) Normal Queries

1. Fetch all products along with their supplier name (INNER JOIN).

2. Find all customers and their orders, even if they have not placed any (LEFT JOIN).

3. Get all suppliers and the products they supply, even if no products exist for a supplier (RIGHT JOIN).

4. Show all customers and all orders (FULL OUTER JOIN simulation using UNION).

5. List all products priced between ₹5000 and ₹50,000 and supplied from "Mumbai".

b) Aggregations & Group By

6. Find the total number of orders placed by each customer and show only those who placed more than 2 (GROUP BY + HAVING).

7. Show each supplier's total sales value (sum of quantity × price_each).

8. Find the average, highest, and lowest price of products in each category.

9. Find the top 5 customers by total spending (ORDER BY SUM(total_amount) DESC LIMIT 5).

10. Show the number of unique products ordered by each customer.

c) Subqueries

11. Find customers who placed an order with an amount greater than the average order amount (subquery).

12. Find products that have never been ordered (subquery with NOT IN).

13. List customers who ordered at least one product from the "Electronics" category.

14. Get suppliers who provide products that have been ordered more than 100 times in total.

15. Find the most expensive product(s) using a subquery with MAX().

d) Advanced Filters

16. Show orders placed by customers who live in either Mumbai, Delhi, or Bengaluru (IN operator).

17. Show orders where payment mode is NOT UPI or Credit Card (NOT IN).

18. Find customers who have no email address recorded (IS NULL).

19. Show suppliers who are not from the same city as any customer (NOT IN subquery).

20. Get the latest 3 orders placed, skipping the first 2 (ORDER BY + LIMIT + OFFSET).

Table creation and Import of data from csv file

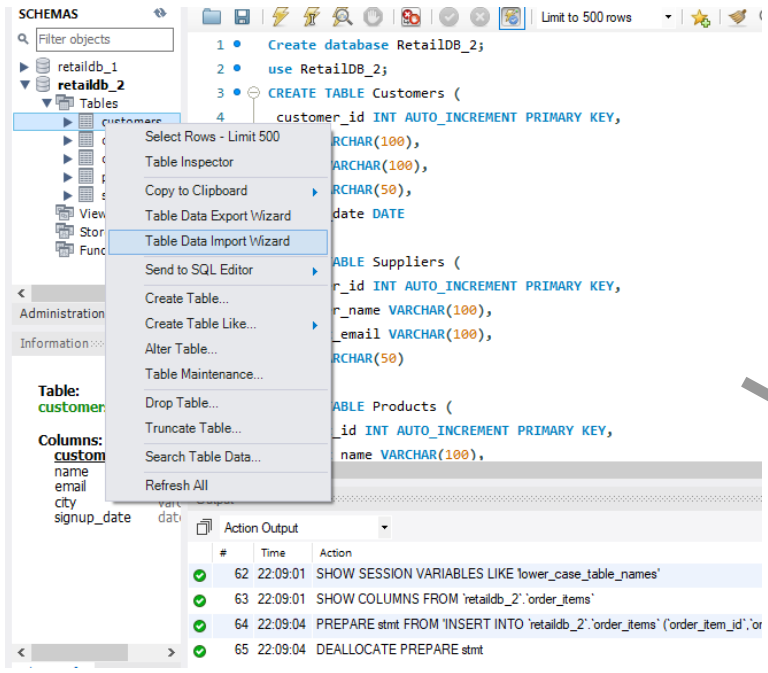
Step 1: Database creation

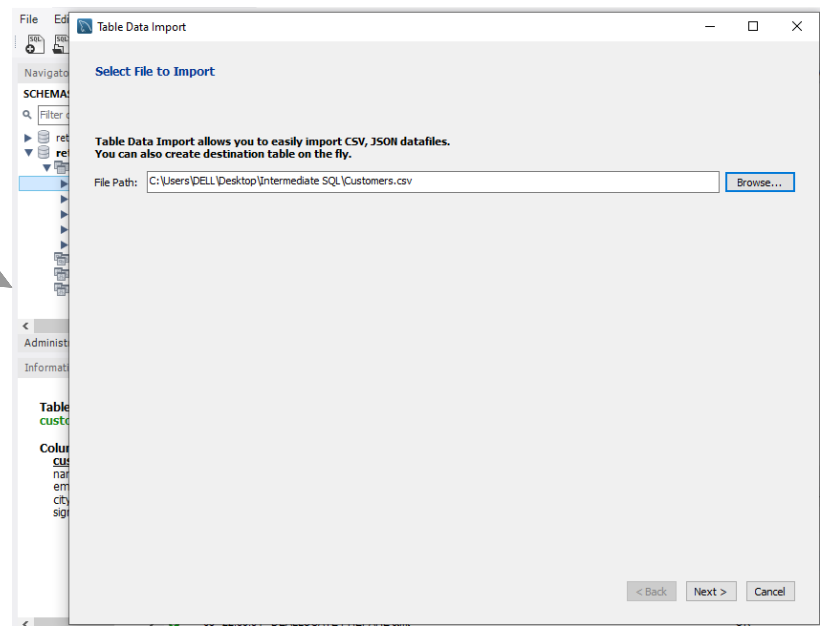
```
1  Create database RetailDB_2;
2  use RetailDB_2;
```

Step 2: Table creation (Customer)

```
CREATE TABLE Customers (
customer_id INT AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(100),
email VARCHAR(100),
city VARCHAR(50),
signup_date DATE
);
```

Step 3: Import data from CSV file





Step 4: Display Imported data

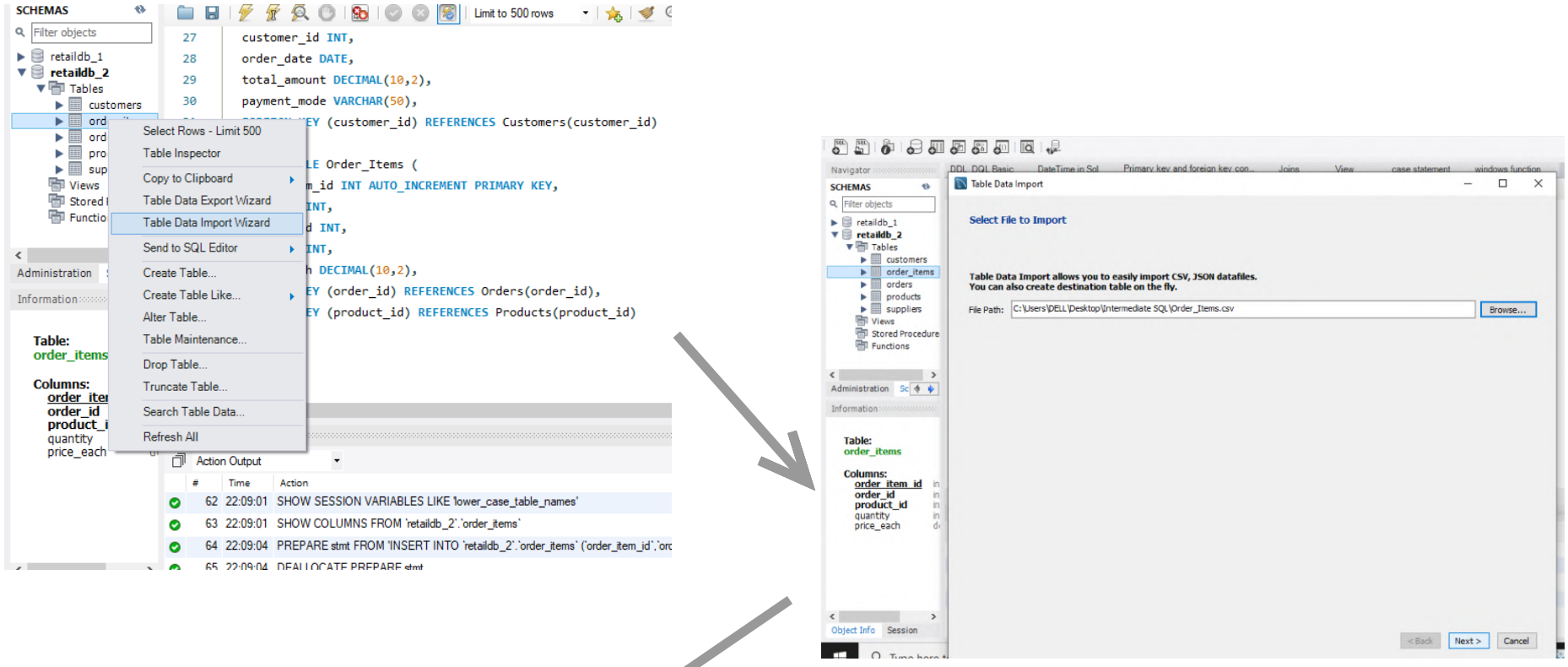
```
select * from customers;
```

customer_id	name	email	city	signup_date
1	Rohit Kumar	rohit.kumar@gmail.com	Delhi	2023-01-12
2	Sneha Sharma	sneha.sharma@yahoo.com	Mumbai	2023-02-05
3	Amit Patel	amit.patel@gmail.com	Ahmedabad	2023-02-18
4	Priya Reddy	priya.reddy@gmail.com	Hyderabad	2023-03-02
5	Karan Singh	karan.singh@outlook.com	Chennai	2023-03-15
6	Neha Verma	neha.verma@gmail.com	Pune	2023-04-01
7	Arjun Mehta	arjun.mehta@gmail.com	Bengaluru	2023-04-20
8	Ritika Gupta	ritika.gupta@yahoo.com	Kolkata	2023-05-12
9	Vikram Joshi	vikram.joshi@gmail.com	Lucknow	2023-05-25

Step 5: Table creation (Orders_Items)

```sql
CREATE TABLE Order_Items (
    order_item_id INT AUTO_INCREMENT PRIMARY KEY,
    order_id INT,
    product_id INT,
    quantity INT,
    price_each DECIMAL(10,2),
    FOREIGN KEY (order_id) REFERENCES Orders(order_id),
    FOREIGN KEY (product_id) REFERENCES Products(product_id)
);
```

Step 6: Import data from CSV file



Step 7: Display Imported data

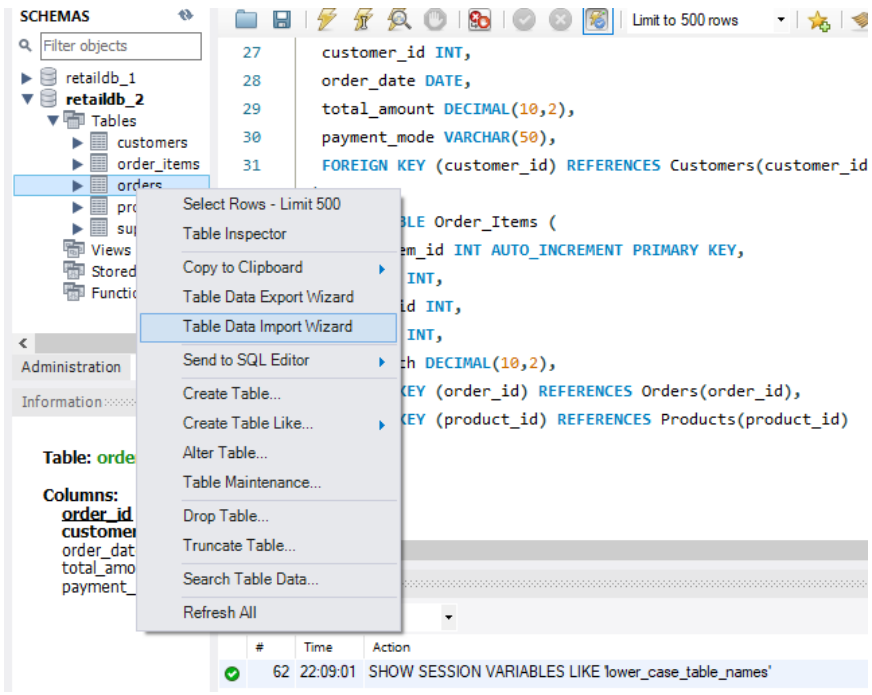
```sql
select * from order_items;
```

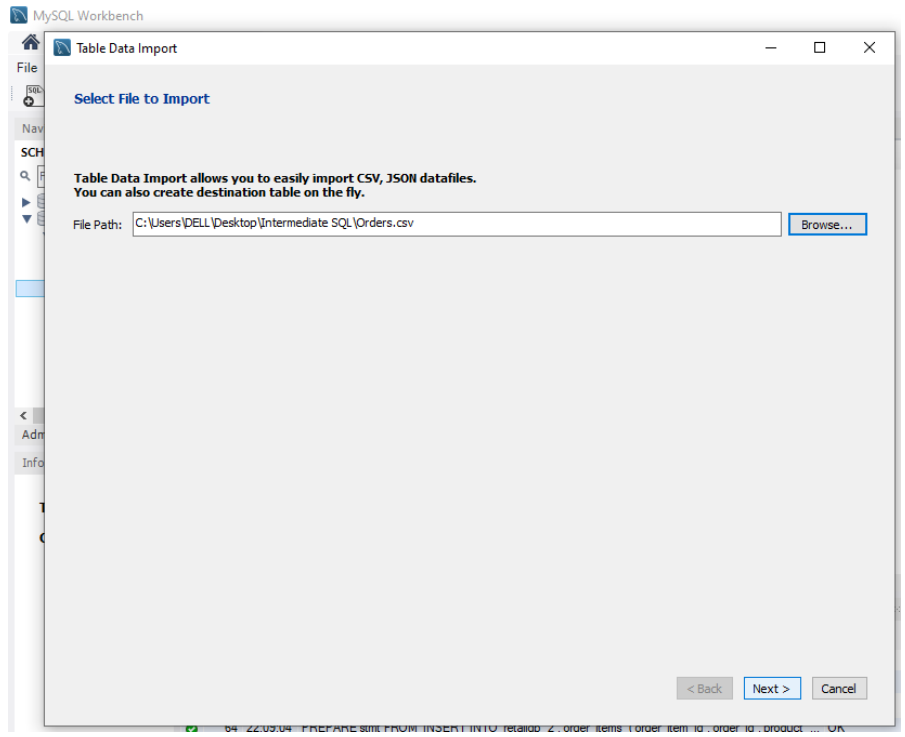
order_item_id	order_id	product_id	quantity	price_each
1	1	1	1	79999.00
2	2	2	1	74999.00
3	3	3	2	4999.00
4	4	4	1	2499.00
5	5	5	3	1999.00
6	6	6	1	2999.00
7	7	7	1	6499.00
8	8	8	1	59999.00
9	9	9	1	55999.00
10	10	10	2	1499.00

Step 8: Table creation (Orders)

```sql
CREATE TABLE Orders (
    order_id INT AUTO_INCREMENT PRIMARY KEY,
    customer_id INT,
    order_date DATE,
    total_amount DECIMAL(10,2),
    payment_mode VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);
```

Step 9: Import data from CSV file





Step 10: Display Imported data

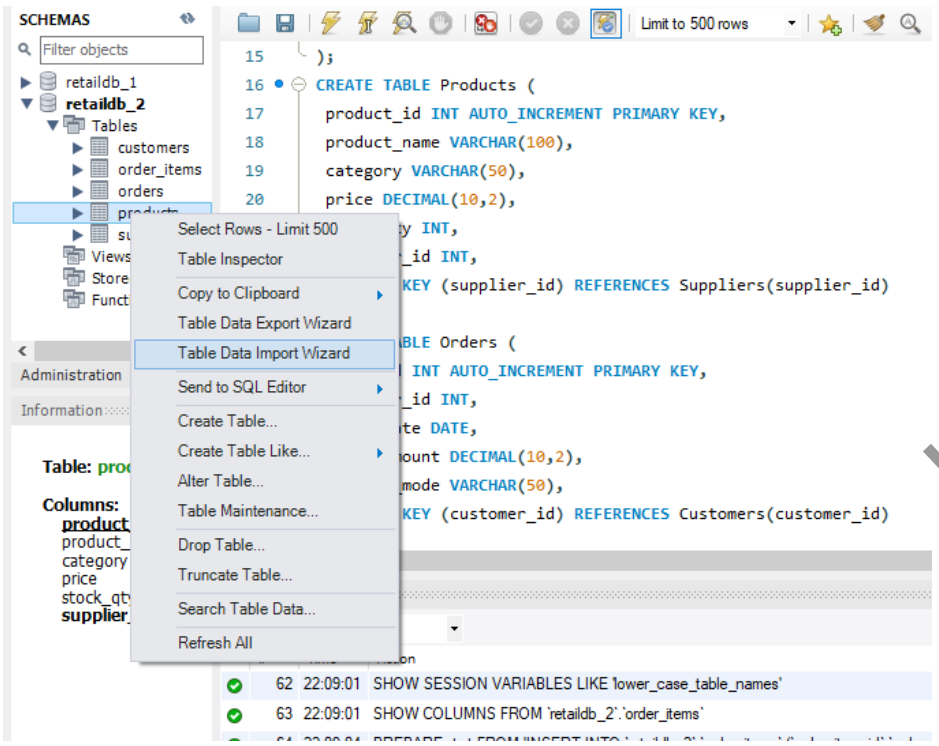
```sql
select * from orders;
```

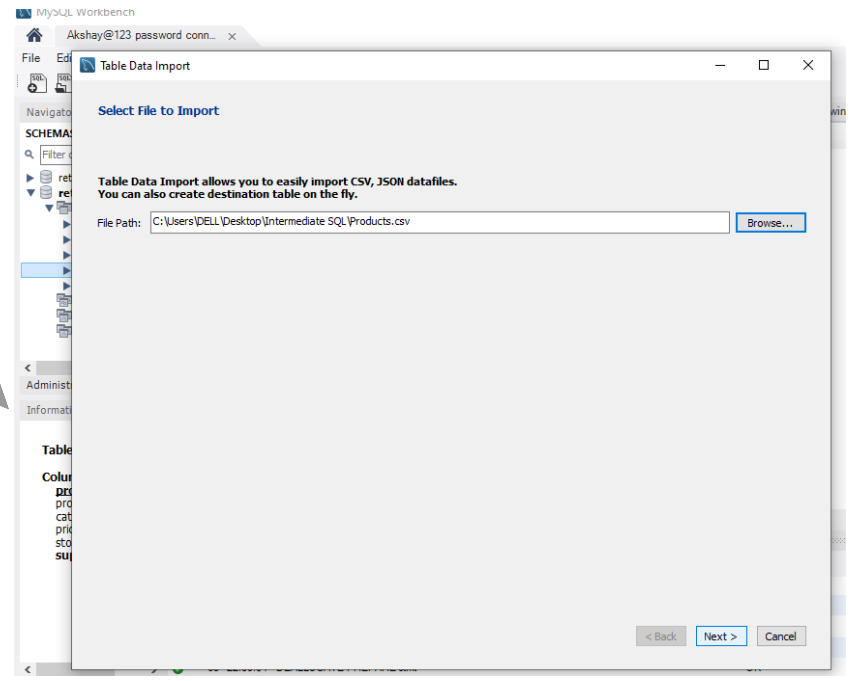
order_id	customer_id	order_date	total_amount	payment_mode
1	1	2024-01-05	79999.00	UPI
2	2	2024-01-08	74999.00	Credit Card
3	3	2024-01-10	9998.00	Debit Card
4	4	2024-01-15	2499.00	UPI
5	5	2024-01-18	5997.00	Cash
6	6	2024-01-20	2999.00	UPI
7	7	2024-01-22	6499.00	Credit Card
8	8	2024-01-25	59999.00	Net Banking
9	9	2024-01-28	55999.00	UPI
10	10	2024-01-30	2998.00	Debit Card

Step 10: Table creation (Products)

```sql
CREATE TABLE Products (
    product_id INT AUTO_INCREMENT PRIMARY KEY,
    product_name VARCHAR(100),
    category VARCHAR(50),
    price DECIMAL(10,2),
    stock_qty INT,
    supplier_id INT,
    FOREIGN KEY (supplier_id) REFERENCES Suppliers(supplier_id)
);
```

Step 11: Import data from CSV file





Step 12: Display Imported data

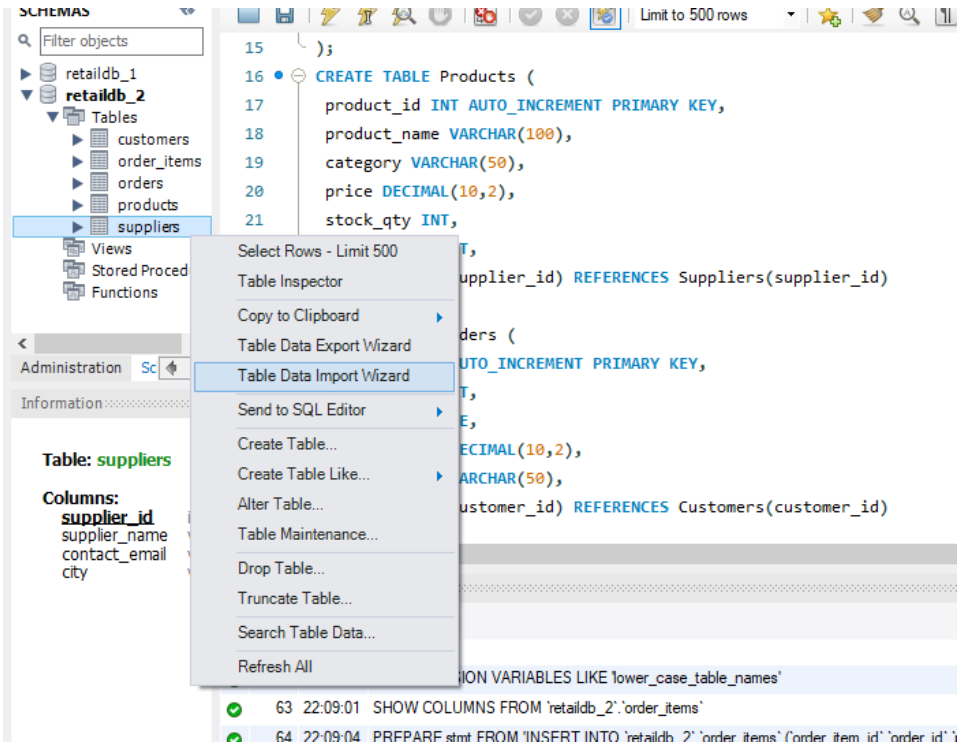
```sql
select * from products;
```

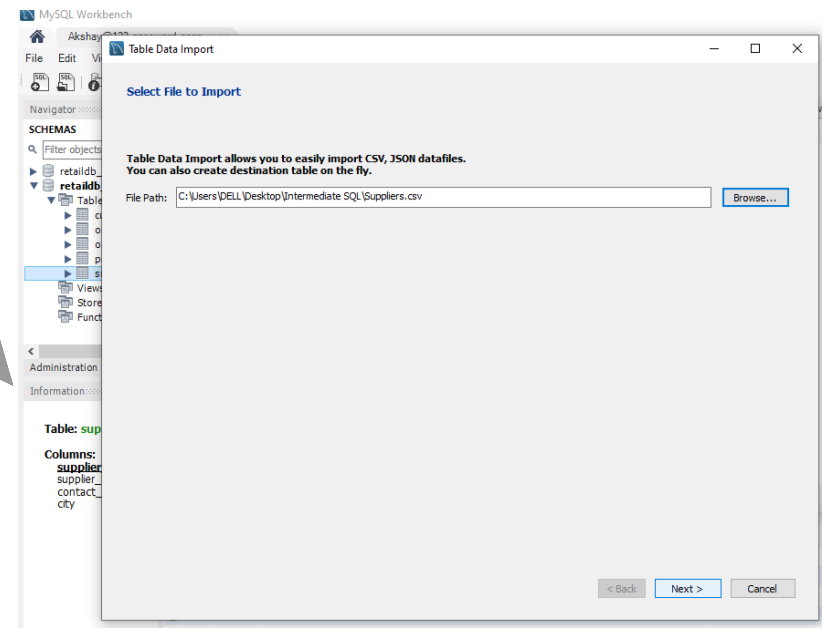
product_id	product_name	category	price	stock_qty	supplier
1	iPhone 15	Electronics	79999.00	50	1
2	Samsung Galaxy S24	Electronics	74999.00	40	2
3	Noise Smartwatch	Wearables	4999.00	100	3
4	Boat Earbuds	Wearables	2499.00	200	4
5	Kurta Set	Fashion	1999.00	150	7
6	Running Shoes	Fashion	2999.00	120	7
7	Prestige Mixer Grinder	Home Appliances	6499.00	80	5
8	Sony Bravia 55? TV	Electronics	59999.00	30	6
9	Lenovo Laptop	Electronics	55999.00	45	2

Step 13: Table creation (Suppliers)

```sql
CREATE TABLE Suppliers (
supplier_id INT AUTO_INCREMENT PRIMARY KEY,
supplier_name VARCHAR(100),
contact_email VARCHAR(100),
city VARCHAR(50)
);
```

Step 14: Import data from CSV file





Step 15: Display Imported data

```sql
select * from suppliers;
```

supplier_id	supplier_name	contact_email	city
1	Reliance Digital	support@reliance.com	Mumbai
2	Croma Electronics	info@croma.com	Delhi
3	Flipkart Seller Hub	sellers@flipkart.com	Bengaluru
4	Amazon India Vendor	vendor@amazon.in	Hyderabad
5	Tata Cliq	support@tatacliq.com	Pune
6	Vijay Sales	sales@vijaysales.com	Chennai
7	Lifestyle Stores	info@lifestylestores.com	Kolkata
8	Metro Electronics	metro@electronics.com	Ahmedabad
NULL	NULL	NULL	NULL

Solved problem statements

1. Fetch all products along with their supplier name (INNER JOIN).

```
select supplier_name from suppliers
inner join products
on suppliers.supplier_id=products.supplier_id;
```

supplier_name
Reliance Digital
Croma Electronics
Croma Electronics
Flipkart Seller Hub
Amazon India Vendor
Tata Cliq
Vijay Sales
Lifestyle Stores
Lifestyle Stores

2. Find all customers and their orders, even if they have not placed any (LEFT JOIN).

```
select * from customers
left join orders
on customers.customer_id=orders.customer_id;
```

customer_id	name	email	city	signup_date	order_id	customer_id	order_date	total_amount	payment_mode
1	Rohit Kumar	rohit.kumar@gmail.com	Delhi	2023-01-12	1	1	2024-01-05	79999.00	UPI
1	Rohit Kumar	rohit.kumar@gmail.com	Delhi	2023-01-12	31	1	2024-03-16	1499.00	Cash
2	Sneha Sharma	sneha.sharma@yahoo.com	Mumbai	2023-02-05	2	2	2024-01-08	74999.00	Credit Card
2	Sneha Sharma	sneha.sharma@yahoo.com	Mumbai	2023-02-05	26	2	2024-03-05	6499.00	Credit Card
2	Sneha Sharma	sneha.sharma@yahoo.com	Mumbai	2023-02-05	45	2	2024-04-13	74999.00	Net Banking
3	Amit Patel	amit.patel@gmail.com	Ahmedabad	2023-02-18	3	3	2024-01-10	9998.00	Debit Card
3	Amit Patel	amit.patel@gmail.com	Ahmedabad	2023-02-18	32	3	2024-03-18	2999.00	UPI
3	Amit Patel	amit.patel@gmail.com	Ahmedabad	2023-02-18	46	3	2024-04-15	5999.00	UPI
4	Priya Reddy	priya.reddy@gmail.com	Hyderabad	2023-03-02	4	4	2024-01-15	2499.00	UPI

3. List all products priced between ₹5000 and ₹50,000 and supplied from "Mumbai".

```
select suppliers.supplier_name from suppliers  join
 products on suppliers.supplier_id = products.supplier_id
 where products.price between 5000 and 100000 and
suppliers.city = 'Mumbai';
```

	supplier_name
▶	Reliance Digital

4. Find the total number of orders placed by each customer and show only those who placed more than 2 (GROUP BY + HAVING).

```
select c.name,count(o.order_id)
from customers as c
join orders as o on
c.customer_id =o.customer_id
group by c.name having  count(o.order_id)>2 ;
```

	name	count(o.order_id)
▶	Sneha Sharma	3
	Amit Patel	3
	Karan Singh	4
	Neha Verma	3
	Arjun Mehta	4
	Ritika Gupta	4
	Vikram Joshi	5
	Ananya Das	3
	Suresh Iyer	3
	Megha Kapoor	3

5. Show each supplier's total sales value (sum of quantity × price_each).

```
select s.supplier_name ,
sum(p.price * ot.quantity) as sale_value
from suppliers as s inner join products as p
on s.supplier_id=p.supplier_id
inner join order_items as ot on p.product_id=ot.product_id
group by s.supplier_name;
```

	supplier_name	sale_value
	Reliance Digital	399995.00
	Croma Electronics	841987.00
	Flipkart Seller Hub	29994.00
	Amazon India Vendor	17493.00
	Tata Cliq	25996.00
	Vijay Sales	239996.00
	Lifestyle Stores	40983.00
▶	Metro Electronics	10493.00

6. Find the average, highest, and lowest price of products in each category.

```
select category ,
avg(price) as price_avg,
max(price) as highest_price
,min(price) as lowest_price
 from products group by category;
```

	category	price_avg	highest_price	lowest_price
▶	Electronics	67749.000000	79999.00	55999.00
	Wearables	3749.000000	4999.00	2499.00
	Fashion	2499.000000	2999.00	1999.00
	Home Appliances	6499.000000	6499.00	6499.00
	Personal Care	1499.000000	1499.00	1499.00

7 . Find the top 5 customers by total spending (ORDER BY SUM(total_amount) DESC LIMIT 5).

```
select c.name,sum(o.total_amount) from customers as c
inner join orders as o on c.customer_id =o.customer_id
group by c.name order by sum(o.total_amount) desc limit 5;
```

	name	sum(o.total_amount)
▶	Ritika Gupta	259996.00
	Vikram Joshi	255994.00
	Aditya Jain	161997.00
	Sneha Sharma	156497.00
	Suresh Iyer	139996.00

8. Find customers who placed an order with an amount greater than the average order amount (subquery).

```
select c.name ,o.total_amount
from  customers as c
inner join orders as o
on c.customer_id =o.customer_id
where o.total_amount >
(select avg(total_amount) from orders );
```

	name	total_amount
▶	Rohit Kumar	79999.00
	Sneha Sharma	74999.00
	Ritika Gupta	59999.00
	Vikram Joshi	55999.00
	Suresh Iyer	79999.00
	Megha Kapoor	74999.00
	Ritika Gupta	59999.00

9. List customers who ordered at least one product from the "Electronics"

```sql
select c.name from customers as c
join orders as o
on c.customer_id = o.customer_id
join order_items as oi
on o.order_id=oi.order_id
join products as p on
pi.product_id =p.product_id
where p.category ="Electronics";
```

name
Ritika Gupta
Vikram Joshi
Vikram Joshi
Ravi Shankar
Neha Verma
Ananya Das
Vikram Joshi

10. Get suppliers who provide products that have been ordered more than 100 times in total.

```sql
SELECT
    s.supplier_id,
    s.supplier_name AS supplier_name,
    SUM(oi.quantity) AS total_ordered
FROM suppliers s
JOIN products p
    ON s.supplier_id = p.supplier_id
JOIN order_items oi
    ON p.product_id = oi.product_id
GROUP BY s.supplier_id, s.supplier_name
HAVING SUM(oi.quantity) > 100;
```

11. Find the most expensive product(s) using a subquery with MAX().

```
use retail_db2;
select * from products;
select product_name,category,
max(price) from products
group by category,product_name
order by max(price) desc
limit 1;
```

	product_name	category	max(price)
▶	iPhone 15	Electronics	79999.00

12 Show orders placed by customers who live in either Mumbai, Delhi, or Bengaluru (IN operator).

```
select * from orders as o join customers as c
on o.customer_id =c.customer_id
where c.city in ("Mumbai","Delhi","Bengluru");
```

	order_id	customer_id	order_date	total_amount	payment_mode	customer_id	name	email	city	signup_date
▶	1	1	2024-01-05	79999.00	UPI	1	Rohit Kumar	rohit.kumar@gmail.com	Delhi	2023-01-12
	31	1	2024-03-16	1499.00	Cash	1	Rohit Kumar	rohit.kumar@gmail.com	Delhi	2023-01-12
	2	2	2024-01-08	74999.00	Credit Card	2	Sneha Sharma	sneha.sharma@yahoo.com	Mumbai	2023-02-05
	26	2	2024-03-05	6499.00	Credit Card	2	Sneha Sharma	sneha.sharma@yahoo.com	Mumbai	2023-02-05
	45	2	2024-04-13	74999.00	Net Banking	2	Sneha Sharma	sneha.sharma@yahoo.com	Mumbai	2023-02-05

13. Show orders where payment mode is NOT UPI or Credit Card (NOT IN).

```
select * from orders
where payment_mode not in ("UPI","Credit Card");
```

order_id	customer_id	order_date	total_amount	payment_mode
3	3	2024-01-10	9998.00	Debit Card
5	5	2024-01-18	5997.00	Cash
8	8	2024-01-25	59999.00	Net Banking
10	10	2024-01-30	2998.00	Debit Card
12	12	2024-02-05	74999.00	Net Banking

14. Find customers who have no email address recorded (IS NULL).

```
select * from customers where email is null;
```

customer_id	name	email	city	signup_date
NULL	NULL	NULL	NULL	NULL

15. Show suppliers who are not from the same city as any customer (NOT IN subquery).

```
select  * from customers as c
join suppliers as s on c.city = s.city ;
```

customer_id	name	email	city	signup_date	supplier_id	supplier_name	contact_email	city
1	Rohit Kumar	rohit.kumar@gmail.com	Delhi	2023-01-12	2	Croma Electronics	info@croma.com	Delhi
2	Sneha Sharma	sneha.sharma@yahoo.com	Mumbai	2023-02-05	1	Reliance Digital	support@reliance.com	Mumbai
3	Amit Patel	amit.patel@gmail.com	Ahmedabad	2023-02-18	8	Metro Electronics	metro@electronics.com	Ahmedabad
4	Priya Reddy	priya.reddy@gmail.com	Hyderabad	2023-03-02	4	Amazon India Vendor	vendor@amazon.in	Hyderabad
5	Karan Singh	karan.singh@outlook.	priya.reddy@gmail.com	-03-15	6	Vijay Sales	sales@vijaysales.com	Chennai

16. Get the latest 3 orders placed, skipping the first 2 (ORDER BY + LIMIT + OFFSET).

```
select * from orders order by order_date asc limit 3 offset 2 ;
```

order_id	customer_id	order_date	total_amount	payment_mode
3	3	2024-01-10	9998.00	Debit Card
4	4	2024-01-15	2499.00	UPI
5	5	2024-01-18	5997.00	Cash
NULL	NULL	NULL	NULL	NULL

17.Get all suppliers and the products they supply, even if no products exist for a supplier (RIGHT JOIN).

```
select * from suppliers
 right join products
on suppliers.supplier_id = products.supplier_id;
```

supplier_id	supplier_name	contact_email	city	product_id	product_name	category	price	stock_qty	supplier_id
1	Reliance Digital	support@reliance.com	Mumbai	1	iPhone 15	Electronics	79999.00	50	1
2	Croma Electronics	info@croma.com	Delhi	2	Samsung Galaxy S24	Electronics	74999.00	40	2
3	Flipkart Seller Hub		Bengaluru	3	Noise Smartwatch	Wearables	4999.00	100	3
4	Amazon India Vendor	vendor@amazon.in	Hyderabad	4	Boat Earbuds	Wearables	2499.00	200	4
7	Lifestyle Stores	info@lifestylestores.com	Kolkata	5	Kurta Set	Fashion	1999.00	150	7

Croma Electronics

18 Show all customers and all orders (FULL OUTER JOIN simulation using UNION).

```
select * from customers left join orders on customers.customer_id=orders.order_id union
select * from customers right join orders on customers.customer_id=orders.order_id;
```

customer_id	name	email	city	signup_date	order_id	customer_id	order_date	total_amount	payment_mode
1	Rohit Kumar	rohit.kumar@gmail.com	Delhi	2023-01-12	1	1	2024-01-05	79999.00	UPI
2	Sneha Sharma	sneha.sharma@yahoo.com	Mumbai	2023-02-05	2	2	2024-01-08	74999.00	Credit Card
3	Amit Patel	amit.patel@gmail.com	Ahmedabad	2023-02-18	3	3	2024-01-10	9998.00	Debit Card
4	Priya Reddy	priya.reddy@gmail.com	Hyderabad	2023-03-02	4	4	2024-01-15	2499.00	UPI
5	Karan Singh	karan.singh@outlook.com	Chennai	2023-03-15	5	5	2024-01-18	5997.00	Cash

19. Find customers who placed an order with an amount greater than the average order amount (subquery).

```sql
select c.name ,o.total_amount
from  customers as c
inner join orders as o
on c.customer_id =o.customer_id
where o.total_amount >
(select avg(total_amount) from orders );
```

	name	total_amount
▶	Rohit Kumar	79999.00
	Sneha Sharma	74999.00
	Ritika Gupta	59999.00
	Vikram Joshi	55999.00
	Suresh Iyer	79999.00

20 .Find products that have never been ordered (subquery with NOT IN).

```sql
select p.product_name
from products as p
where p.product_id
not in (select order_id from orders);
```

Overview of MySQL workbench

```sql
select supplier_name from suppliers
inner join products
on suppliers.supplier_id=products.supplier_id;
```

```sql
#2. Find all customers and their orders, even if they have not placed any (LEFT JOIN).

select * from customers
left join orders
on customers.customer_id=orders.customer_id;
```

```sql
#3. Get all suppliers and the products they supply, even if no products exist for a supplier (F
select * from suppliers right join products on suppliers.supplier_id = products.supplier_id;
```

```sql
#4. Show all customers and all orders (FULL OUTER JOIN simulation using UNION).
select * from customers left join orders on customers.customer_id=orders.order_id union
select * from customers right join orders on customers.customer_id=orders.order_id;
```

```sql
#5. List all products priced between ₹5000 and ₹50,000 and supplied from "Mumbai".
select suppliers.supplier_name from suppliers  join
 products on suppliers.supplier_id = products.supplier_id
 where products.price between 5000 and 100000 and
suppliers.city = 'Mumbai';
```

```sql
#6. Find the total number of orders placed by each customer and show only those who placed more

select c.name,count(o.order_id)
from customers as c
join orders as o on
c.customer_id =o.customer_id
group by c.name having  count(o.order_id)>2 ;
```

```sql
#7. Show each supplier's total sales value (sum of quantity × price_each).
select s.supplier_name ,
sum(p.price * ot.quantity) as sale_value
from suppliers as s inner join products as p
on s.supplier_id=p.supplier_id
inner join order_items as ot on p.product_id=ot.product_id
group by s.supplier_name;
```

```sql
#8. Find the average, highest, and lowest price of products in each category.

select category ,
avg(price) as price_avg,
max(price) as highest_price
,min(price) as lowest_price
 from products group by category;
```

```sql
#9. Find the top 5 customers by total spending (ORDER BY SUM(total_amount) DESC LIMIT 5).
select c.name,sum(o.total_amount) from customers as c
inner join orders as o on c.customer_id =o.customer_id
group by c.name order by sum(o.total_amount) desc limit 5;
```

```sql
#10. Show the number of unique products ordered by each customer.
select c.name , count(distinct p.category )  as p_category from
customers as c inner join products as p on c.customer_id = p.supplier_id
group by c.name;
```

```sql
#11. Find customers who placed an order with an amount greater than the average order amount
select c.name ,o.total_amount
from  customers as c
inner join orders as o
on c.customer_id =o.customer_id
where o.total_amount >
(select avg(total_amount) from orders );
```

```sql
#12. Find products that have never been ordered (subquery with NOT IN).

select p.product_name
from products as p
where p.product_id
not in (select order_id from orders);
```

```sql
select p.product_name
from products as p inner join orders as o
on p.product_id= o.customer_id
where o.customer_id
 not in (select customer_id from orders);
```



```sql
#13. List customers who ordered at least one product from the "Electronics" category.
select c.name from customers as c
join orders as o
on c.customer_id = o.customer_id
join order_items as oi
on o.order_id=oi.order_id
join products as p on
oi.product_id =p.product_id
where p.category ="Electronics";
```

```sql
#15. Find the most expensive product(s) using a subquery with MAX().
use retail_db2;
select * from products;
select product_name,category,
max(price) from products
group by category,product_name
order by max(price) desc
limit 1;
```

```sql
# 16. Show orders placed by customers who live in either Mumbai, Delhi, or Bengaluru (IN operat
select * from orders;
select* from customers;
select * from orders as o join customers as c on o.customer_id =c.customer_id where c.city in (
```

```sql
#17. Show orders where payment mode is NOT UPI or Credit Card (NOT IN).
select * from orders where payment_mode not in ("UPI","Credit Card");
```

```sql
#18. Find customers who have no email address recorded (IS NULL).
select * from customers;
select * from customers where email is null;
```

```sql
#19. Show suppliers who are not from the same city as any customer (NOT IN subquery).
select * from customers;
select * from suppliers;
select  * from customers as c join suppliers as s on c.city = s.city ;
```

Thank

you